Evacuating Robots from a Disk Using Face-to-Face Communication ⋆ ⋆⋆

J. Czyzowicz,[1], K. Georgiou,[2], E. Kranakis[3], L. Narayanan[4], J. Opatrny[4], and B. Vogtenhuber[5]

[1] Dépt. d'Informatique, Université du Québec en Outaouais, Gatineau, QC, Canada
Jurek.Czyzowicz@uqo.ca
[2] Dept. of Comb. & Opt., University of Waterloo, Waterloo, ON, Canada
k2georgiou@uwaterloo.ca
[3] School of Computer Science, Carleton University, Ottawa, ON, Canada
kranakis@scs.carleton.ca
[4] Dept. of Comp. Science and Soft. Engineering, Concordia University, Montreal, QC, Canada
lata, opatrny@cs.concordia.ca
[5] Institute for Softwaretechnology, Graz University of Technology, Graz, Austria
bvogt@ist.tugraz.at

Abstract. Assume that two robots are located at the centre of a unit disk. Their goal is to *evacuate* from the disk through an *exit* at an unknown location on the boundary of the disk. At any time the robots can move anywhere they choose on the disk, independently of each other, with maximum speed 1. The robots can cooperate by exchanging information whenever they meet. We study algorithms for the two robots to minimize the *evacuation time*: the time when *both* robots reach the exit.
In [9] the authors gave an algorithm defining trajectories for the two robots yielding evacuation time at most 5.740 and also proved that any algorithm has evacuation time at least $3+\frac{\pi}{4}+\sqrt{2}\approx 5.199$. We improve both the upper and lower bounds on the evacuation time of a unit disk. Namely, we present a new non-trivial algorithm whose evacuation time is at most 5.628 and show that any algorithm has evacuation time at least $3+\frac{\pi}{6}+\sqrt{3}\approx 5.255$. To achieve the upper bound, we designed an algorithm which non-intuitively proposes a forced meeting between the two robots, even if the exit has not been found by either of them.

1 Introduction

The goal of traditional search problems is to find an object which is located in a specific domain. This subject of research has a long history and there is a plethora of models investigated in the mathematical and theoretical computer science literature with emphasis on probabilistic search in [16], game theoretic applications in [3], cops and robbers in [8], classical pursuit and evasion in [15], search problems and group testing in [1], and many more.

In this paper, we investigate the problem of searching for a stationary point target called an *exit* at an unknown location using two robots. This type of collaborative search is advantageous in that it reduces the required search time by distributing the search effort between the two robots. In previous work on collaborative search, the goal has generally been to minimize the time taken by the *first* robot to find the object of the search. In contrast, in this work, we are interested in minimizing the time when the *last robot* finds the exit. In particular, suppose two robots are in the interior of a region with a single exit. The robots need to evacuate the region but the location of the exit is unknown to them. The robots can cooperate to search for the exit, but it is not enough for one robot to find the exit, we require *both* robots to reach the exit as soon as possible.

⋆ This work was partially supported by NSERC grants
⋆⋆ An extended abstract of this work is accepted for publication in the LNCS proceedings of the 9th International Conference on Algorithms and Complexity (CIAC15).

We study the problem of two robots that start at the same time at the centre of a unit disk and attempt to reach an exit placed at an unknown location on the boundary of the disk. At any time the robots can move anywhere they choose within the disk. Indeed, they can take short-cuts by moving in the interior of the disk if desired. We assume their maximum speed is 1. The robots can communicate with each other only if they are at the same point at the same time: we call this communication model *face-to-face communication*. Our goal is to schedule the trajectories of the robots so as to minimize the *evacuation time*, which is the time it takes both robots to reach the exit (for the worst case location of the exit).

1.1 Related work

The most related work to ours is [9], where the evacuation problem for a set of robots all starting from the centre of a unit disk was introduced and studied. Two communication models are introduced in [9]. In the *wireless* model, the two robots can communicate at any time regardless of their locations. In particular, a robot that finds the exit can immediately communicate its location to the other robot. The other model is called the *non-wireless or local* model in [9], and is the same as our face-to-face model: two robots can only communicate when they are face to face, that is, they are at the same point location at the same time. In [9], for the case of 2 robots, an algorithm with evacuation time $1 + \frac{2\pi}{3} + \sqrt{3} \approx 4.826$ is given for the wireless model; this is shown to be optimal. For the face-to-face model, they prove an upper bound of 5.740 and a lower bound of 5.199 on the evacuation time.

Baeza-Yates *et al* posed the question of minimizing the worst-case trajectory of a single robot searching for a target point at an unknown location in the plane [4]. This was generalized to multiple robots in [14], and more recently has been studied in [11,13]. However, in these papers, the robots cannot communicate, and moreover, the objective is for the first robot to find the target. Two seminal and influential papers (that appeared almost at the same time) on probabilistic search are [5], and [6] and concern minimizing the *expected time* for the robot to find the target. Useful surveys on search theory can also be found in [7] and [10]. In addition, the latter citation has an interesting classification of search problems by search objectives, distribution of effort, point target (stationary, large, moving), two-sided search, etc. The evacuation problem considered in our paper is related to searching on a line, in that we are searching on the boundary of a disk but with the additional ability to make short-cuts in order to enable the robots to meet sooner and thus evacuate faster.

Our problem is also related to the *rendezvous problem* and the problem of *gathering* [2,12]. Indeed our problem can be seen as a version of a rendezvous problem for three robots, where one of them remains stationary.

1.2 Preliminaries and notation

We assume that two robots R_1 and R_2 are initially at the center of a disk with radius 1, and that there is an exit at some location X on the boundary of the disk. The robots do not know X, but do know each other's algorithms. The robots move at a speed subject to a maximum speed, say 1. They cannot communicate except if they are at the same location at the same time. Finally, both robots are equipped with deterministic processors that can numerically solve trigonometric equations, and as such they are assumed to have the required memory. The *evacuation problem* is to define trajectories for the two robots that minimize the *evacuation time*.

For two points A and B on the unit circle, the length of an arc AB is denoted by $\widehat{\mathrm{AB}}$, while the length of the corresponding chord (line segment) will be denoted by $\overline{AB}$ (arcs on the circle are always read clockwise, i.e., arc AB together with arc BA cover the whole circle). By $A\hat{B}C$ we denote the angle at B in the triangle ABC. Finally by $\overrightarrow{AB}$ we denote the vector with tail A and tip B.

1.3 Outline and results of the paper

In [9] an algorithm is given defining a trajectory for two robots in the face-to-face communication model with evacuation time 5.740 and it is also proved that any such algorithm has evacuation time at least $3 + \frac{\pi}{4} + \sqrt{2} > 5.199$.

Our main contribution in this paper is to improve both the upper and lower bounds on the evacuation time. Namely, we give a new algorithm whose evacuation time is at most 5.628 (see Section 2) and also prove that any algorithm has evacuation time at least $3 + \frac{\pi}{6} + \sqrt{3} > 5.255$ (see Section 3). To prove our lower bound on the disk, we first give tight bounds for the problem of evacuating a regular hexagon where the exit is placed at an unknown vertex. We observe that, surprisingly, in our optimal evacuation algorithm for the hexagon, the two robots are forced to meet after visiting a subset of vertices, even if an exit has not been found at that time. We use the idea of such a forced meeting in the design of our disk evacuation algorithm.

2 Evacuation Algorithms

In this section we give two new evacuation algorithms for two robots in the face-to-face model that take evacuation time approximately 5.644 and 5.628 respectively. We begin by presenting Algorithm $\mathcal{A}$ proposed by [9] which has been shown to have evacuation time 5.740. Our goal is to understand the worst possible configuration for this algorithm, and subsequently to modify it accordingly so as to improve its performance.

All the algorithms we present follow the same general structure: The two robots R_1 and R_2 start by moving together to an arbitrary point A on the boundary of the disk. Subsequently R_1 explores the arc $A'A$, where A' is the antipodal point of A, by moving along some trajectory defined by the algorithm. At the same time, R_2 explores the arc AA', following a trajectory that is the reflection of R_1's trajectory. If either of the robots finds the exit, it immediately uses the *Meeting Protocol* defined below to meet the other robot (note that the other robot has not yet found the exit and hence keeps exploring). After meeting, the two robots travel together on the shortest path to the exit, thereby completing the evacuation. At all times, the two robots travel at unit speed. Without loss of generality, we assume that R_1 finds the exit and then *catches* R_2 for our analysis.

Meeting Protocol for R_1 *: If at any time* t_0 R_1 *finds the exit at point* X*, it computes the shortest additional time* t *such that* R_2*, after traveling distance* $t_0 + t$*, is located at point* M *satisfying* $\overline{XM} = t$*. Robot* R_1 *moves along the segment* XM*. At time* $t_0 + t$ *the two robots meet at* M *and traverse directly back to the exit at* X *incurring total time cost* $t_0 + 2t$*.*

2.1 Evacuation Algorithm $\mathcal{A}$ of [9]

We proceed by describing the trajectories of the two robots in Algorithm $\mathcal{A}$. As mentioned above, both robots start from the centre O of the disk and move together to an arbitrary position A on

the boundary of the disk. R_2 then moves clockwise along the boundary of the disk up to distance π, see left-hand side of Figure 1, and robot R_1 moves counter clockwise on the trajectory which is a reflection of R_2's trajectory with respect to the line passing through O and A. When R_1 finds the exit, it invokes the *meeting protocol* in order to meet R_2, after which the evacuation is completed.

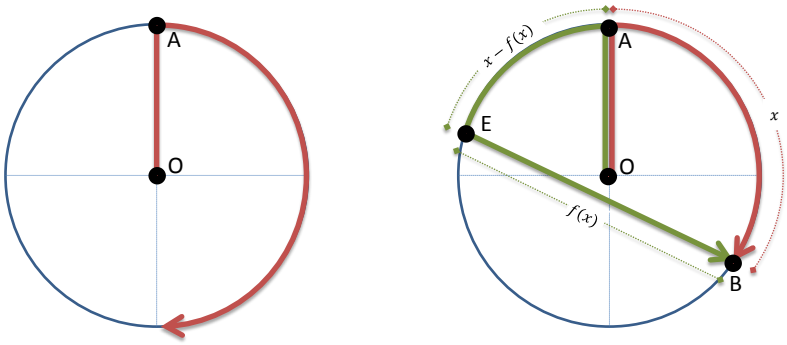

Fig. 1. Evacuation Algorithm $\mathcal{A}$ with exit position E. The trajectory of robot R_2 is depicted on the left. The movement paths of robots R_1, R_2 are shown on the right, till the moment they meet at point B on the circle.

The meeting-protocol trajectory of R_1 in Algorithm $\mathcal{A}$ is depicted in the right-hand side of Figure 1. Clearly, for the two robots to meet, we must have $\overset{\frown}{\mathrm{AB}} = \overset{\frown}{\mathrm{EA}} + \overline{EB}$. Next we want to analyze the performance of the algorithm, with respect to $x := \overset{\frown}{\mathrm{AB}}$, i.e. the length x of the arc that R_2 travels, before it is met by R_1. We also set $f(x) := \overline{EB}$.

It follows that $\overset{\frown}{\mathrm{EA}} = x - f(x)$, and since $2\sin(\overset{\frown}{\mathrm{EB}}/2) = \overline{EB}$ we conclude that

$$f(x) \;=\; z \;\text{ where } z \text{ is a solution of the equation } z = 2\sin\left(x - \frac{z}{2}\right). \tag{1}$$

In other words, $f(x)$ is the length of interval EB that R_1 needs to travel in the interior of the disk after locating the exit at E, to meet R_2 at point B.

Then, the cost of Algorithm $\mathcal{A}$, given that the two robots meet at time x after they together reached the boundary of the disk at A, is $1 + x + f(x)$. Given that distance $x - f(x)$ traveled by R_1 until finding the exit is between 0 and π, it directly follows that x can take any value between 0 and π as well. Hence, the worst case performance of Algorithm $\mathcal{A}$ is determined by $\sup_{x\in[0,\pi]}\{x + f(x)\}$ The next lemma, along with its proof, follows from [9].

Lemma 1. *Expression $F(x) := x + f(x)$ attains its supremum at $x_0 \approx 2.85344$ (which is $\approx 0.908279\pi$). In particular, $F(x)$ is strictly increasing when $x \in [0, x_0]$ and strictly decreasing when $x \in [x_0, \pi]$*

Proof (Sketch). The behavior of $F(x)$, as x ranges in $[0, \pi]$, is shown in Figure 2.

By Lemma 1, the evacuation time of Algorithm $\mathcal{A}$ is $1 + x_0 + f(x_0) < 5.740$. The worst case is attained for $x_0 - f(x_0) \approx 0.308\pi$.

2.2 New evacuation algorithm $\mathcal{B}(\chi, \phi)$

We now show how to improve the previously described algorithm and obtain evacuation time at most 5.644. The main idea for improving Algorithm $\mathcal{A}$ is to change the trajectory of the robots

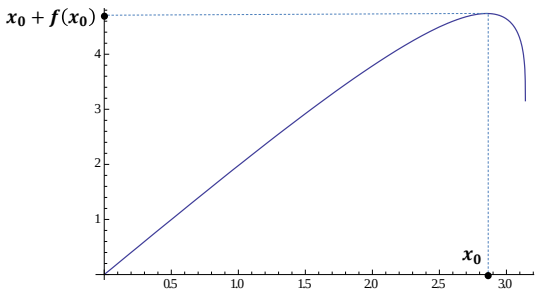

Fig. 2. The performance of Algorithm $\mathcal{A}$ as a function of the meeting points of the robots.

when the distance traveled on the boundary of the disk approaches the critical value x_0 of Lemma 1. Informally, robot R_2 could meet R_1 earlier if it makes a linear detour inside the interior of the disk towards R_1 a little before traversing distance x_0.

We describe a generic family of algorithms that realizes this idea. The specific trajectory of each algorithm is determined by two parameters χ and ϕ where $\chi \in [\pi/2, x_0]$ and $\phi \in [0, f(\chi)/2]$, whose optimal values will be determined later. For ease of exposition, we assume R_1 finds the exit. The trajectory of R_2 (assuming it has not yet met R_1) is partitioned into four phases that we call the *deployment*, *pre-detour*, *detour* and *post-detour* phases. The description of the phases rely on the left-hand side of Figure 3.

Algorithm $\mathcal{B}(\chi, \phi)$(with a linear detour). R_2's trajectory until it meets R_1 is described below:
$\star$ *Deployment phase:* Robot R_2 starts from the centre O of the disk and moves to an arbitrary position A on the boundary of the disk.
$\star$ *Pre-detour phase:* R_2 moves clockwise along the boundary of the disk until having explored an arc of length χ.
$\star$ *Detour phase:* Let D be the reflection of B with respect to AA' (where A' is the antipodal point of A). Then, R_2 moves on a straight line towards the interior of the disk and towards the side where O lies, forming an angle of ϕ with line BD, until R_2 touches line AA' at point C. From C it follows a straight line segment to B. Note that C is indeed in the interior of the line segment AA' by the restrictions on ϕ.
$\star$ *Post-detour phase:* Robot R_2 continues moving clockwise on the arc BA'.
At the same time R_1 follows a trajectory that is the reflection of R_2's trajectory along the line AA'. When at time t_0 it finds the exit, it follows the Meeting Protocol defined above.

Notably, the two robots may meet at point C without having located the exit. Next we consider three cases as to where R_2 can be caught by R_1 while moving on its trajectory (after R_1 has located the exit). For all three cases, the reader can consult the right-hand side of Figure 3. As the time needed for the deployment phase is independent of where the exit is located, we ignore this extra cost of 1 during the case distinction.

Case 1: R_2 is caught during its pre-detour phase: The meeting point is anywhere on the arc AB. Recall that $\chi \leq x_0$, so by Lemma 1 the location F of the exit on the arc FA that maximizes the cost of $\mathcal{B}(\chi, \phi)$ is the one at at distance $\chi - f(\chi)$ from A (see right-hand side of Figure 3). The cost then is $\overset{\frown}{\mathrm{AB}} + \overline{BF} = \overset{\frown}{\mathrm{FA}} + 2\overline{BF} = \chi + f(\chi)$.

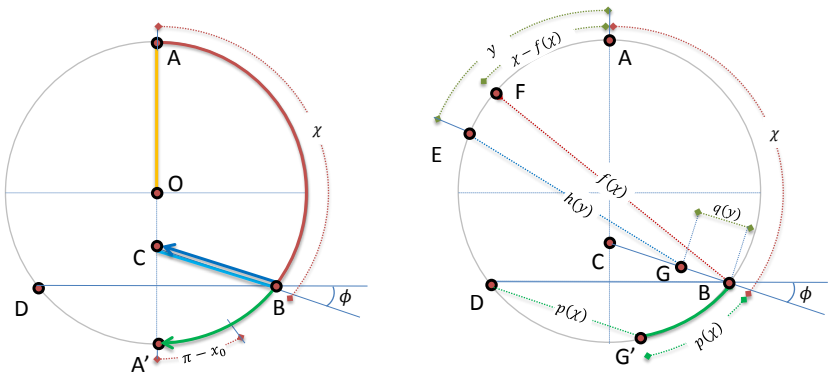

Fig. 3. Illustrations for Algorithm $\mathcal{B}(\chi, \phi)$.

Case 2: R_2 is caught during its detour phase: Let G be the point on BC where the robots meet. Further, let E be the position of the exit on the arc $A'A$, and let $y := \widehat{\mathrm{EA}}$. In the following, $h(y) := \overline{EG}$ denotes the length of the trajectory of R_1 in its attempt to catch R_2 after it finds the exit. Also, $q(y) := \overline{BG}$ denotes the distance that R_2 travels on BC till it is caught by R_1. Note that the functions h and q also depend on χ and ϕ; however, while those are fixed, y varies with the position of the exit. Lemma 2 below states that $h(y)$ and $q(y)$ are well defined.

Lemma 2. *$y \in [\chi - f(\chi), \chi]$ if and only if the meeting point G of the robots is on the line segment BC. Moreover, robot R_2 can be caught by R_1 only while moving from B to C.*

Proof. If the exit is located at point F (i.e. $y = \chi - f(\chi)$), then the meeting point is B. We also observe that if the exit-position coincides with D (i.e. $y = \chi$), then the meeting point is C. Recall that $\phi \leq f(\chi)/2 = \widehat{\mathrm{DF}}/2$, and hence point C is in the interior or on the boundary of the triangle FDB as it is depicted in right-hand side of Figure 3. Therefore, after time χ, robot R_2 would approach the exit if it was located anywhere on the arc DE. In particular, if $y \in [\chi - f(\chi), \chi]$ then the meeting point for Algorithm $\mathcal{A}$ would be on the arc BA'. In Algorithm $\mathcal{B}(\chi, \phi)$, R_2 has a trajectory that brings it closer to the exit. This guarantees that the meeting point G always exists, and it lies in the line segment BC (we will soon derive a closed formula relating $\widehat{\mathrm{EA}}$ and $\overline{BG}$). The previous argument guarantees that $h(y)$ is continuous and strictly decreasing in y. Notice that $\widehat{\mathrm{EA}} + \overline{EG} = \widehat{\mathrm{AB}} + \overline{BG}$ (since the two robots start from the same position A), which means that

$$y + h(y) = \chi + q(y) \tag{2}$$

Hence, $q(y) = y + h(y) - \chi$, and as already explained, $q(\chi - f(\chi)) = 0$ and $q(\chi) = \overline{BC}$. By the mean value theorem, all values between 0 and $\overline{BC}$ are attainable for $q(y)$ and are attained while y ranges in $[\chi - f(\chi), \chi]$. ∎

We conclude that if the exit is located at point E, then the cost of the algorithm is $y + 2h(y)$. Hence, in case 2, the cost of the algorithm is at most $\sup_{y \in [\chi - f(\chi), \chi]} \{y + 2h(y)\}$. We emphasize again that $h(y)$ and $q(y)$ also depend on the fixed parameters χ and ϕ.

Case 3: R_2 is caught during its post-detour phase: Clearly, in this case the exit lies in the interior of the arc $A'D$ or coincides with A'. At time $t_d = \chi + 2q(\chi)$, robots R_1 and R_2 are located at points D and B, respectively. Then they move towards each other on the arc BD till R_1 finds the exit. Note that, since $\overline{DB}/2 = \sin(\chi)$, we have $q(\chi) = \sin(\chi) / \cos(\phi)$. Clearly, the closer the exit is to D is, the higher is the cost of the evacuation algorithm. In the limit (as

the position of the exit approaches D), the cost of case 3 approaches t_d plus the time it takes R_1 to catch R_2 if the exit was located at D, and if they started moving from points D and B respectively. Let G' be the meeting point on the arc BD in this case, i.e. $\overline{DG'} = \overset{\frown}{\text{BG'}}$. We define $p(x)$ to be the distance that R_1 needs to travel in the interior of the disk to catch R_2, if the exit is located at distance x from A. Clearly[6]

$$p(x) \;:=\; \text{unique } z \text{ satisfying } z = 2\sin\left(x + \frac{z}{2}\right). \tag{3}$$

Note also that $\overline{DG'} = p(\chi)$ so that the total cost in this case is at most

$$t_d + 2p(\chi) = \chi + 2\sin(\chi)/\cos(\phi) + 2p(\chi).$$

The following two lemmata summarize the above analysis and express $h(y)$ in explicit form (in dependence of χ and ϕ), respectively.

Lemma 3. *The evacuation time of Algorithm $\mathcal{B}(\chi,\phi)$ is*

$$1 + \max\left\{\chi + f(\chi), \sup_{y\in[\chi-f(\chi),\chi]}\{y+2h(y)\}, \quad \chi + 2\sin(\chi)/\cos(\phi) + 2p(\chi)\right\}, \tag{4}$$

where $h(y)$ (that also depends on the choice of χ,ϕ) denotes the time that a robot needs from the moment it finds the exit till it meets the other robot when following the meeting protocol.

Lemma 4. *For every $\chi > 0$ and for every $\chi - f(\chi) \le y \le \chi$, the distance $h(y)$ that R_1 travels from A until finding R_2 when following the meeting protocol in Algorithm $\mathcal{B}(\chi,\phi)$ is*

$$h(y) = \frac{2 + (\chi-y)^2 - 2\cos(\chi+y) + 2(\chi-y)\left(\sin(\phi+y) - \sin(\phi-\chi)\right)}{2(\chi - y - \sin(\phi-\chi) + \sin(\phi+y))}.$$

In particular, $h(y)$ is strictly decreasing for $0 \le \phi \le f(\chi)/2$.

Proof. We start by making some handy observations. For this we rely on Figure 4 (that is a continuation of Figure 3). Let H be the point that is symmetric to E with respect to AA'. Denote with L the projection of H onto the supporting line of DB. Set $\theta := B\hat{H}L$, and observe the following equation for θ.

$$\begin{aligned}\theta &= \frac{\pi}{2} - E\hat{H}B = \frac{\pi}{2} - \frac{\overset{\frown}{\text{BE}}}{2} = \frac{\pi}{2} - \frac{\overset{\frown}{\text{BD}} + \overset{\frown}{\text{DE}}}{2} \\ &= \frac{\pi}{2} - \frac{2(\pi-\chi)+\chi-y}{2} = \frac{\chi+y}{2} - \frac{\pi}{2}\end{aligned} \tag{5}$$

Our goal is to compute $h(y) = \overline{EG}$. For this we see that $\overrightarrow{EG} = \overrightarrow{EH} + \overrightarrow{HB} + \overrightarrow{BG}$, and therefore

$$\overline{EG}^2 = \overline{EH}^2 + \overline{HB}^2 + \overline{BG}^2 + 2\left(\overrightarrow{EH}\cdot\overrightarrow{HB} + \overrightarrow{EH}\cdot\overrightarrow{BG} + \overrightarrow{HB}\cdot\overrightarrow{BG}\right). \tag{6}$$

[6] Uniqueness of the root of the equation defining $p(x)$ is an easy exercise.

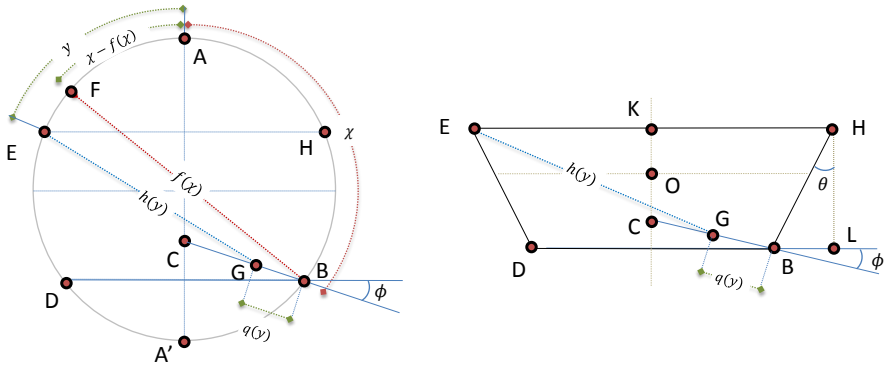

Fig. 4. The analysis of Algorithm $\mathcal{B}(\chi, \phi)$.

We have $\overline{EH} = 2\sin(y)$, $\overline{HB} = 2\sin\left(\frac{\chi - y}{2}\right)$, $\overline{BG} = q(y)$, and

$$\begin{aligned}
\overrightarrow{EH} \cdot \overrightarrow{HB} &= \overline{EH}\ \overline{HB} \cos\left(E\hat{H}B\right) \\
\overrightarrow{EH} \cdot \overrightarrow{BG} &= \overline{EH}\ \overline{BG} \cos\left(G\hat{B}L\right) \\
\overrightarrow{HB} \cdot \overrightarrow{BG} &= \overline{HB}\ \overline{BG} \cos\left(\pi - G\hat{B}H\right).
\end{aligned}$$

We also have

$$\begin{aligned}
\cos\left(E\hat{H}B\right) &= \cos(\pi/2 + \theta) = \cos\left(\frac{\chi + y}{2}\right) \\
\cos\left(G\hat{B}L\right) &= \cos(\pi - \phi) = \cos(\phi) \\
\cos\left(\pi - G\hat{B}H\right) &= \cos\left(\pi + \frac{\pi}{2} + \theta - \phi\right) = -\cos\left(\frac{\chi + y}{2} - \phi\right).
\end{aligned}$$

Substituting the above in (6), we obtain an equation between $h(y)$, $q(y)$, y, χ, and ϕ. In the latter equation we can substitute $q(y)$ using the meeting condition (2), obtaining this way the required closed formula for $h(y)$. ■

The first natural attempt in order to beat Algorithm $\mathcal{A}$ would be to consider $\mathcal{B}(\chi, 0)$, i.e. make BC perpendicular to AA' in Figure 3. In light of Lemma 4 and using Lemma 3, we state the following claim to build some intuition for our next, improved, algorithm.

Claim 1 *The performance of algorithm $B(\chi, 0)$ is optimized when $\chi = \chi_0 \approx 2.62359$, and its cost is $1 + 4.644 = 5.644$. The location of the exit inducing the worst case for $B(\chi, 0)$ is when $y = \widehat{EA} \approx 0.837\pi$. The meeting point of the two robots takes place at point G (see Figure 3, and set $\phi = 0$), where $q(y) = \overline{BG} \approx 0.117 \approx 0.236\overline{BC}$. In particular, the cost of the algorithm, if the meeting point of the robots is during R_2's pre-detour, detour and post-detour phase, is (approximately) 5.621, 5.644 and 5.644 respectively.*

Note that χ_0 of Claim 1 is strictly smaller than x_0 of Lemma 1. In other words, the previous claim is in coordination with our intuition that if the robots moved towards the interior of the disk a little before the critical position of the meeting point x_0 of Algorithm $\mathcal{A}$, then the cost of the algorithm could be improved.

2.3 New evacuation algorithm $\mathcal{C}(\chi, \phi, \lambda)$

Claim 1 is instructive for the following reason. Note that the worst meeting point G for Algorithm $\mathcal{B}(\chi_0, 0)$ satisfies $\overline{BG} \approx 0.236\overline{BC}$. This suggests that if we consider algorithm $\mathcal{B}(\chi_0, \phi)$ instead, where $\phi > 0$, then we would be able to improve the cost if the meeting point happened during the detour phase of R_2. On one hand, this further suggests that we can decrease the detour position χ_0 (note that the increasing in χ cost $\chi + f(\chi)$ is always a lower bound to the performance of our algorithms when $\chi < x_0$). On the other hand, that would have a greater impact on the cost when the meeting point is in the post-detour phase of R_2, as in this case the cost of moving from B to C and back to B would be $2\sin(\chi)/\cos(\phi)$ instead of just $2\sin(\chi)$. A compromise to this would be to follow the linear detour trajectory of R_2 in $\mathcal{B}(\chi_0, \phi)$ only up to a certain threshold-distance λ, after which the robot should reach the diameter segment AA' along a linear segment perpendicular to segment AA' then return to the detour point B along a linear segment. Thus the detour forms a triangle. This in fact completes the high level description of Algorithm $\mathcal{C}(\chi, \phi, \lambda)$ that we formally describe below.

In that direction, we fix χ, ϕ, λ, with $\chi \in [\pi/2, x_0], \phi \in [0, f(\chi)/2]$ and $\lambda \in [0, \sin(\chi)/\cos(\phi)]$. As before, we only describe the trajectory of robot R_2. The meeting protocol that R_1 follows once it finds the exit is the same as for Algorithms $\mathcal{A}$ and $\mathcal{B}(\chi, \phi)$.

The trajectory of robot R_2 (that has neither found the exit nor met R_1 yet) can be partitioned into roughly the same four phases as for Algorithm $\mathcal{B}(\chi, \phi)$; so we again call them *deployment*, *pre-detour*, *detour* and *post-detour* phases. The description of the phases refers to the left-hand side of Figure 5, which is a partial modification of Figure 3.

Algorithm $\mathcal{C}(\chi, \phi, \lambda)$ (with a triangular detour). The phases of robot R_2's trajectory are:
⋆ *Deployment phase:* Same as in Algorithm $\mathcal{B}(\chi, \phi)$. At time 1, R_2 is at point A.
⋆ *Pre-detour phase:* Same as in Algorithm $\mathcal{B}(\chi, \phi)$. In additional time χ, R_2 is in point B.
⋆ *Detour phase:* This phase is further split into three subphases.
◇ *Subphase-1:* Up to additional time λ, R_2 moves along a line segment exactly as in the detour phase of Algorithm $\mathcal{B}(\chi, \phi)$. Let G be the position of the robot at the end of this phase.
◇ *Subphase-2:* Let C be the projection of G onto AA'. R_2 follows line segment GC till it reaches point C.
◇ *Subphase-3 (Recovering phase):* Robot follows line segment CB back to point B.
⋆ *Post-detour phase:* Same as in Algorithm $\mathcal{B}(\chi, \phi)$. After additional time $\overset{\frown}{\mathrm{EA}}$', R_2 reaches point A'.
At the same time R_1 follows a trajectory that is the reflection of R_2's trajectory along the line AA'. If a robot finds the exit, it follows the meeting protocol defined earlier.

Obviously, Algorithm $\mathcal{C}\left(\chi, \phi, \frac{\sin(\chi)}{\cos(\phi)}\right)$ is identical to Algorithm $\mathcal{B}(\chi, \phi)$. Moreover, as before robots may meet at point C without having located the exit.

Notice that an immediate consequence of the definition of $\mathcal{C}(\chi, \phi, \lambda)$ is that if robot R_1 finds the exit and meets R_2 during its detour subphase-2 in some point K (as in the right-hand side of Figure 5), then

$$\overset{\frown}{\mathrm{EA}} + \overline{EK} = \overset{\frown}{\mathrm{AB}} + \overline{BG} + \overline{GK}. \tag{7}$$

When R_1 finds the exit somewhere on the arc $A'A$, it catches R_2 on its trajectory so that they return together to the exit. Note that since robots meet at point C, if the exit is not in the arc DB, it is impossible for a robot to be caught by the other robot in subphase-3 of its detour phase.

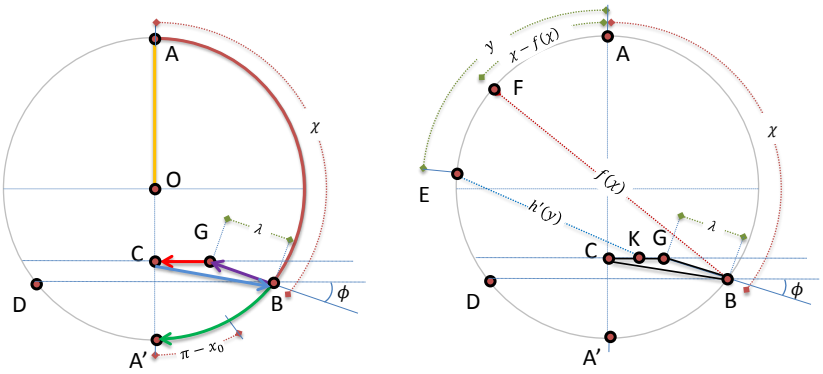

Fig. 5. Illustrations for Algorithm $\mathcal{C}(\chi, \phi, \lambda)$.

Hence, there are four cases as to where R_2 can be caught by R_1 that found the exit. As before, we omit the extra cost 1 which is the time needed for the deployment phase during the case distinction.

- Case 1: R_2 is caught in its pre-detour phase: The cost of Algorithm $\mathcal{C}(\chi, \phi, \lambda)$ is at most $\chi + f(\chi)$, exactly as in the analogous case of Algorithm $\mathcal{B}(\chi, \phi)$.
- Case 2: R_2 is caught in its detour subphase-1: As in case 2 of the analysis of Algorithm $\mathcal{B}(\chi, \phi)$, if E is the position of the exit then, $y = \widehat{\mathrm{EA}}$ satisfies $y \geq \chi - f(\chi)$ (the relevant figure in this case remains the left-hand side of Figure 4). As longs as $q(y)$ remains less than λ, the cost of the algorithm remains $y + 2h(y)$. In order to find the maximum y for which this formula remains valid, we need to solve the equation $\lambda = q(y)$. This is possible by recalling that $h(y) = \chi + q(y) - y$, and by invoking the formula of $h(y)$ as it appears in Lemma 4. By the monotonicity of $h(y)$, we have that there exists unique ψ satisfying $h(\psi) = \chi + \lambda - \psi$. It follows that the cost of the algorithm in this case is at most $\sup_{\chi - f(\chi) \leq y \leq \psi} \{y + 2h(y)\}$.
- Case 3: R_2 is caught in its detour subphase-2: In this case, the relevant figure is the right-hand side of Figure 5. Let the exit be at point E, and let K denote the meeting point of the robots on the line segment GC. We set $h'(y) := \overline{EK}$, which is calculated next in Lemma 5 (a). We conclude that in this case the cost of the algorithm is at most $\sup_{\psi \leq y \leq \chi} \{y + 2h'(y)\}$.
- Case 4: R_2 is caught in its post-detour phase: Let t_d again be the total time a robot needs till it enters its post-detour phase. As in case 3 of Algorithm $\mathcal{B}(\chi, \phi)$, the cost of Algorithm $C(\chi\phi, \lambda)$ for this case is at most $t_d + 2p(\chi)$. It thus remains to show how to calculate $t_d = \widehat{\mathrm{AB}} + \overline{BG} + \overline{GC} + \overline{CB}$, which is done in Lemma 5 (b).

Lemma 5. *The following statements hold for Algorithm $\mathcal{C}(\chi, \phi, \lambda)$:*
(a) Suppose that R_1 finds the exit and meets R_2 in its detour subphase-2. Then the time $h'(y)$ that R_1 needs from finding the exit until meeting R_2 is $h'(y) = N(\chi, y, \lambda, \phi)/D(\chi, y, \lambda, \phi)$, where

$$\begin{aligned} N(\chi, y, \lambda, \phi) := \quad & 2 + \lambda^2 + (\lambda + \chi - y)^2 + 2\lambda(\sin(\phi - \chi) - \sin(\phi + y)) \\ & + 2(\lambda + \chi - y)(\sin(\chi) + \sin(y) - \lambda\cos(\phi)) - 2\cos(\chi + y), \\ D(\chi, y, \lambda, \phi) := & 2(\lambda + \chi - y + \sin(\chi) + \sin(y) - \lambda\cos(\phi)). \end{aligned}$$

(b) Suppose that R_1 finds the exit and meets R_2 in its post-detour phase. Then, the total time that R_2 spends in its detour phase is

$$\lambda + \sin(\chi) - \lambda\cos(\phi) + \sqrt{\sin^2(\chi) + \lambda^2 \sin^2(\phi)}$$

Proof. As an illustration of the proof we refer to Figure 6, which is a continuation of Figure 5. Let H be the symmetric point of E with respect to diameter AA'. As in Figure 4, L is the projection of H onto the supporting line of DB, and θ denotes the angle $B\hat{H}L$, whose value is given by (5). Also G' and C' are the projections of G and C, respectively, onto DL. The calculations below follow the spirit of the arguments in Lemma 4.

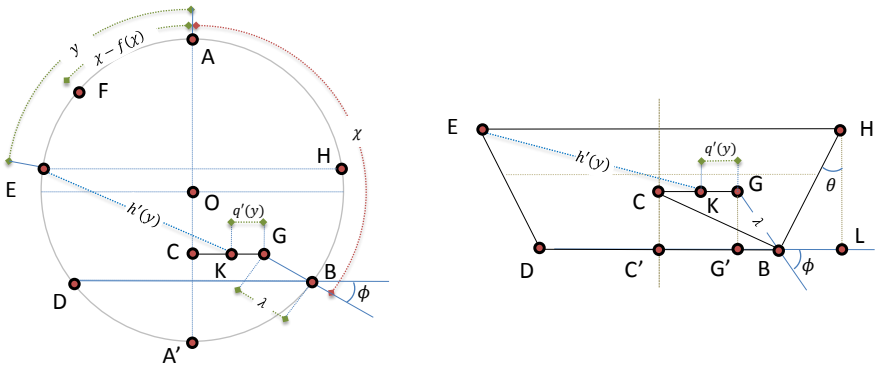

Fig. 6. The analysis of Algorithm $\mathcal{C}(\chi, \phi, \lambda)$.

(a) As before, y denotes the distance of the exit from point A. We have that $\overrightarrow{EK} = \overrightarrow{EH} + \overrightarrow{HB} + \overrightarrow{BG} + \overrightarrow{GC}$, and therefore

$$\begin{aligned}\overline{EK}^2 = \; & \overline{EH}^2 + \overline{HB}^2 + \overline{BG}^2 + \overline{GC}^2 + \\ & 2\left(\overrightarrow{EH}\cdot\overrightarrow{HB} + \overrightarrow{EH}\cdot\overrightarrow{BG} + \overrightarrow{EH}\cdot\overrightarrow{GC} + \overrightarrow{HB}\cdot\overrightarrow{BG} + \overrightarrow{HB}\cdot\overrightarrow{GC} + \overrightarrow{BG}\cdot\overrightarrow{GC}\right), \end{aligned} \tag{8}$$

where $\overline{EK} = h'(y)$, $\overline{EH} = 2\sin(y)$, $\overline{HB} = 2\sin\left(\frac{\chi - y}{2}\right)$, $\overline{BG} = \lambda$, and $\overline{GK} = q'(y)$. The inner products $\overrightarrow{EH}\cdot\overrightarrow{HB}, \overrightarrow{EH}\cdot\overrightarrow{BG}, \overrightarrow{HB}\cdot\overrightarrow{BG}$ are calculated exactly as in Lemma 4. For the remaining inner products we see that

$$\begin{aligned}\overrightarrow{EH}\cdot\overrightarrow{GK} &= \overline{EH}\;\overline{GK}\cos(\pi) = -\overline{EH}\;\overline{GK}, \\ \overrightarrow{HB}\cdot\overrightarrow{GK} &= \overline{HB}\;\overline{GK}\cos\left(\frac{\pi}{2} - \theta\right) = \overline{HB}\;\overline{GK}\sin(\theta), \\ \overrightarrow{BG}\cdot\overrightarrow{GK} &= \overline{BG}\;\overline{GK}\cos(\phi).\end{aligned}$$

Substituting the above in (8), we obtain an equation for $h'(y)$ as a function of $q'(y)$, y, χ, and ϕ. In the latter equation we can substitute $q'(y)$ using the meeting condition (7), according to which $q'(y) = y + h'(y) - \chi - \lambda$. Resolving the resulting equation for $h'(y)$ gives the desired formula.

(b) We need to calculate $\overline{BG} + \overline{GC} + \overline{CB}$, where $\overline{BG} = \lambda$. First, we observe that $\overline{GC} = \overline{G'C'} = \overline{BC'} - \overline{BG'} = \sin(\chi) - \lambda\cos(\phi)$. In order to calculate $\overline{CB}$, we see that $\overrightarrow{BC} = \overrightarrow{BG} + \overrightarrow{GC}$. Hence we obtain

$$\begin{aligned}\overline{BC}^2 &= \overline{BG}^2 + \overline{GC}^2 + 2\overrightarrow{BG}\cdot\overrightarrow{GC} \\ &= \lambda^2 + (\sin(\chi) - \lambda\cos(\phi))^2 + 2\lambda(\sin(\chi) - \lambda\cos(\phi))\cos(\phi) \\ &= \sin^2(\chi) + \lambda^2\sin^2(\phi),\end{aligned}$$

which concludes our claim. ∎

Before stating our main theorem, we summarize the total time required by Algorithm $\mathcal{C}(\chi, \phi, \lambda)$ in the following lemma.

Lemma 6. *The cost of Algorithm $\mathcal{C}(\chi, \phi, \lambda)$ can be expressed as*

$$1 + \max \left\{ \begin{array}{ll} \chi + f(\chi) & \textit{(pre-detour phase)} \\ \sup_{\chi - f(\chi) \le y \le \psi} \{y + 2h(y)\} & \textit{(detour subphase-1)} \\ \sup_{\psi \le y \le \chi} \{y + 2h'(y)\} & \textit{(detour subphase-2)} \\ \chi + \lambda + \sin(\chi) - \lambda \cos(\phi) + \sqrt{\sin^2(\chi) + \lambda^2 \sin^2(\phi) + 2p(\chi)} & \\ & \textit{(post-detour phase)} \end{array} \right\},$$

where the functions f and p are as in (1) and (3), respectively; functions $h(y)$ and $h'(y)$ are expressed explicitly in Lemmas 4 and 5 (a), respectively; and ψ is the unique solution to the equation $h(\psi) = \chi + \lambda - \psi$.

Using the statement of Lemma 6 and numerical optimization, we obtain the following improved upper bound.

Theorem 1. *For $\chi_0 = 2.631865, \phi_0 = 0.44916$ and $\lambda_0 = 0.05762$, the evacuation algorithm $\mathcal{C}(\chi_0, \phi_0, \lambda_0)$ has cost no more than 5.628.*

Proof. We examine the cost of our algorithm depending on where the meeting point of the two robots occurs. The guidelines of the analysis are suggested by Lemma 6. Also the deployment cost of 1 will be added at the end. Any calculations below are numerical, and were performed using MATHEMATICA.

For the given parameters, we see that $f(\chi_0) = 1.99603, p(\chi_0) = 0.506932, \chi_0 - f(\chi_0) = 0.63584$, and $\psi = 0.755204$. If the meeting point is during the pre-detour phase, then the cost is $\chi_0 + f(\chi_0) < 4.62791$ (note that $\chi_0 < x_0$). If the meeting point is in the post-detour phase, then the cost is

$$\chi_0 + \lambda_0 + \sin(\chi_0) - \lambda \cos(\phi_0) + \sqrt{\sin^2(\chi_0) + \lambda_0^2 \sin^2(\phi_0)} + 2p(\chi_0) < 4.627965.$$

For the more interesting intermediate cases, we see that

$$h(y) = \frac{-0.5y^2 + 3.45042y + a(y) + b(y) - 6.61768}{y - 0.900812 \sin(y) - 0.434209 \cos(y) - 3.45042}$$
$$h'(y) = \frac{-0.5y^2 + 3.12552y + (y - 3.12552) \sin(y) - 0.847858 \cos(y) - 5.74387}{y - \sin(y) - 3.12552}$$

where $a(y) := (0.900812y - 2.85875) \sin(y)$ and $b(y) := (0.434209y - 2.01566) \cos(y)$. We can see then that the cost in the detour subphase-2 is

$$\sup_{0.63584 \le y \le 0.755204} \{y + 2h(y)\} < 4.627972,$$

and the cost in the detour subphase-3 is

$$\sup_{0.755204 \le y \le 2.631865} \{y + 2h'(y)\} < 4.627961.$$

This completes the proof of the theorem. ∎

3 Lower Bound

In this section we show that any evacuation algorithm for two robots in the face-to-face model takes time at least $3 + \frac{\pi}{6} + \sqrt{3} \approx 5.255$. We first prove a result of independent interest about an evacuation problem on a hexagon.

Theorem 2. *Consider a hexagon of radius* 1 *with an exit placed at an unknown vertex. The worst case evacuation time for two robots starting at any two arbitrary vertices of the hexagon is at least* $2 + \sqrt{3}$.

Proof. Assume an arbitrary deterministic algorithm $\mathcal{D}$ for the problem. $\mathcal{D}$ solves the problem for any input, i.e., any placement of the exit. We construct two inputs for $\mathcal{D}$ and show that for at least one of them, the required evacuation time is at least $2+\sqrt{3}$. First, we let $\mathcal{D}$ run without placing an exit at any vertex, so as to find out in which order the robots are exploring all the vertices of the hexagon. We label the vertices of the hexagon according to this order (if two vertices are explored simultaneously then we just order them arbitrarily). Let t be the time when the fifth vertex, v_5, of

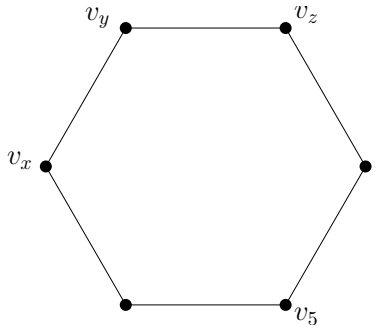

Fig. 7. Vertices of the hexagon as visited by algorithm $\mathcal{D}$; t is the time when the fifth vertex v_5 is visited by some robot, say R_1. One of the vertices adjacent to v_5 has not been visited yet by a robot.

the hexagon is visited by some robot, say R_1, i.e., robot R_1 is at vertex v_5 at time t, and four more vertices of the hexagon have been already visited. (see Figure 7). In other words, v_5 and v_6 are the only vertices[7] that are guaranteed to not have been explored at time $t - \varepsilon$, for any sufficiently small $\varepsilon > 0$. Note that we must have $t \geq 2$, since at least one of the two robots must have visited at least three vertices by this time (and hence must have walked at least the two segments between the first and the second, and between the second and the third vertices visited in its trajectory).

The first input I_1 we construct has the exit placed at vertex v_6. If $t \geq 1 + \sqrt{3}$, then this input gives an evacuation time of at least $2 + \sqrt{3}$. Indeed, until robot R_1 reaches v_5, the algorithm $\mathcal{D}$ processes I_1 identically to the case where there is no exit; further, at time t, robot R_1 needs additional time at least 1 just to reach the exit.

Hence assume that $2 \leq t < 1+\sqrt{3}$. Let v_x, v_y, and v_z be the three vertices that are non-adjacent to v_5 in the hexagon (see Figure 7). Note that the minimum distance between v_5 and any of v_x, v_y, and v_z is at least $\sqrt{3}$. If $v_6 \in \{v_x, v_y, v_z\}$ then on input I_1, $\mathcal{D}$ needs evacuation time at least $t + \sqrt{3} \geq 2 + \sqrt{3}$, as R_1 still has to reach the exit.

[7] It might be that v_4 and v_5 are explored simultaneously, or that v_5 and v_6 are explored simultaneously. In the former case v_6 is explored strictly after v_5 while in the latter v_4 is explored strictly before v_5.

Therefore, assume that $v_6 \notin \{v_x, v_y, v_z\}$ and hence $\{v_x, v_y, v_z\} \subset \{v_1, v_2, v_3, v_4\}$. Note that, since $t < 1 + \sqrt{3}$, on input I_1, robot R_1 has visited at most one of v_x, v_y, and v_z at time t. Hence, the other robot, R_2 has visited at least two of them. For the second input I_2 that we construct, we place the exit on the vertex v^* that is the last vertex among v_x, v_y, and v_z in the visiting order of the vertices by R_2. Let t^* be the time when R_2 reaches v^*, and note that at least until t^*, the algorithm $\mathcal{D}$ behaves identical on the two inputs I_1 and I_2. As R_2 has visited at least one vertex before visiting v^*, we have $t^* \geq 1$.

Next we claim that R_1 and R_2 cannot meet between time t^* and t. The claim below shows that the former is impossible. Namely, we can prove:

Claim 2 *If $t < 1 + \sqrt{3}$, then on input I_2, R_1 and R_2 do not meet between time t^* (defined in the preceding paragraph) and time t.*

Proof. (Claim 2) Assume on the contrary, that on input I_2, R_1 and R_2 do meet at some time t' at point P, with $t^* \leq t' < t$. Observe that on input I_2, robot R_1 continues until time t' as on input I_1 but having met R_2 at time t' might continue differently after time t'. Let $t_B = t' - t^*$ be the time that R_2 uses on input I_2 to get from the exit v^* to P, and let $t_A = t - t'$ be the time that R_1 uses on input I_1 to get to vertex v_5 from P. As v^* and v_5 are at distance at least $\sqrt{3}$, and since $t^* \geq 1$, we have

$$\sqrt{3} \leq t_A + t_B = t - t' + t' - t^* = t - t^* \leq t - 1.$$

So we obtain $\sqrt{3} + 1 \leq t$, which contradicts the assumption $t < 1 + \sqrt{3}$. This proves the claim. ■

Having proved the claim, we conclude that on input I_2, R_1 continues until time t as on input I_1. Hence R_1 needs at least $t + \sqrt{3} \geq 2 + \sqrt{3}$ time to reach the exit on input I_2. This completes the proof of the theorem. ■

It is worth noting that the lower bound from Theorem 2 matches the upper bound of evacuating a regular hexagon, when the initial starting vertices may be chosen by the algorithm. Consider a hexagon $ABCDEF$ and suppose that the trajectory of one robot, as long as no exit was found, is $ABDC$. Similarly, the other robot follows the symmetric trajectory $FECD$; cf. left-hand side of Fig. 8. By symmetry it is sufficient to consider exits at vertices A, B or C. An exit at C is reached by each robot independently, while both robots proceed to an exit at A or B after meeting at point M, the intersection of segments BD and EC. Altogether, they need a total time of at most $\max\{1 + 4/\sqrt{(3)},\ 1 + (2 + \sqrt{7})/\sqrt{3},\ 1 + \sqrt{3} + 1\}$ to evacuate from the hexagon. It is easy then to verify that, in each case, the evacuation time of this algorithm is always upper bounded by $2 + \sqrt{3}$.

In the above algorithm, the robots meet at M, regardless of whether the exit has been already found or not. The idea of our algorithm for disk evacuation presented in the previous section was influenced by this non-intuitive presence of a forced meeting.

Combining Theorem 2 with some reasoning from measure theory, we obtain the following lower bound for our evacuation problem.

Theorem 3. *Assume you have a unit disk with an exit placed somewhere on the boundary. The worst case evacuation time for two robots starting at the centre in the face-to-face model is at least $3 + \frac{\pi}{6} + \sqrt{3} \approx 5.255$.*

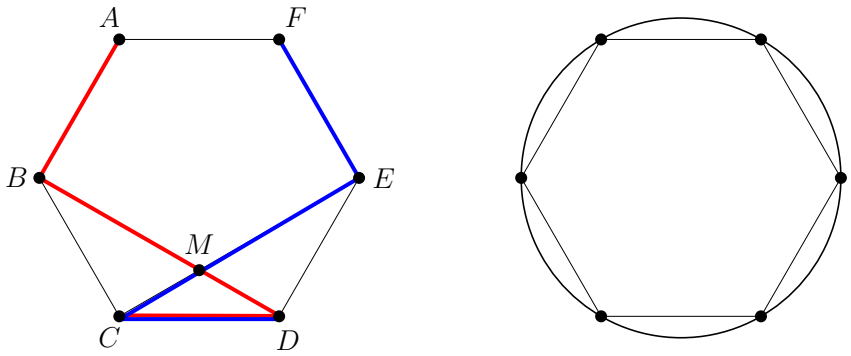

Fig. 8. The trajectories for R_1 (red) and R_2 (blue) for the hexagon evacuation algorithm having evacuation time $2+\sqrt{3}$, while the exit has not been found, are depicted on the left. Right-hand side: At time $1+\frac{\pi}{6}-\varepsilon$, there is regular hexagon all of whose vertices are unexplored and lie on the boundary of the disk.

Proof. It takes 1 time unit for the robots to reach the boundary of the hexagon. By time $t = 1+\frac{\pi}{6}$, any algorithm could have explored at most $\frac{2\pi}{6}$ of the boundary of the disk. Hence for any ε with $0 < \varepsilon < t$, there exists a regular hexagon with all vertices on the boundary of the disk and all of whose vertices are unexplored at time $t-\varepsilon$; see the right-hand side of Figure 8. Now, invoking Theorem 2 gives the bound of at least $1+\frac{\pi}{6}+2+\sqrt{3}$ to evacuate both robots. ∎

4 Conclusion

In this paper we studied evacuating two robots from a disk, where the robots can collaborate using face-to-face communication. Unlike evacuation for two robots in the wireless communication model, for which the tight bound $1+\frac{2\pi}{3}+\sqrt{3}$ is proved in [9], the evacuation problem for two robots in the face-to-face model is much harder to solve. We gave a new non-trivial algorithm for the face-to-face communication model which improved the upper bound in [9]. We used a novel, non-intuitive idea of a forced meeting between the robots, regardless of whether the exit was found before the meeting. We also provided a different analysis that improved the lower bound in [9].

We believe that none of our bounds are close to be tight. More specifically, we do know that our upper bound is not optimal, since by disallowing robots to meet without having found the exit (by slightly truncating their trajectory), we can provably improve the performance of our algorithm. Unfortunately, the improvement we obtain this way is negligible (affecting the third significant decimal digit) while the additional required technicalities would be overwhelming, without offering new insights for the problem. This also suggests that the choice of the parameters we choose for our algorithm are not optimal. We are also certain that the proposed algorithm, i.e. family of trajectories we consider, cannot give the optimal trajectory, as it is intuitive that the optimal solution should be related to a properly defined differential equation ensuring that if robots meet during the deployment phase then the overall cost stays constant. Similarly for the lower bound, we believe that our proposed technique will serve as a guideline towards a more refined analysis that would reduce the gap. To conclude, a tight bound still remains elusive.

Acknowledgements. This work was initiated during the 13^{th} *Workshop on Routing* held in July 2014 in Querétaro, México.

References

1. R. Ahlswede and I. Wegener. *Search problems*. Wiley-Interscience, 1987.
2. S. Alpern and A. Beck. Asymmetric rendezvous on the line is a double linear search problem. *Mathematics of Operations Research*, 24(3):604–618, 1999.
3. S. Alpern and S. Gal. *The theory of search games and rendezvous*, volume 55. Springer, 2003.
4. R. Baeza Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 106(2):234–252, 1993.
5. A. Beck. On the linear search problem. *Israel Journal of Mathematics*, 2(4):221–228, 1964.
6. R. Bellman. An optimal search. *Siam Review*, 5(3):274–274, 1963.
7. S. Benkoski, M. Monticino, and J. Weisinger. A survey of the search theory literature. *Naval Research Logistics (NRL)*, 38(4):469–494, 1991.
8. A. Bonato and R. Nowakowski. *The game of cops and robbers on graphs*. American Mathematical Soc., 2011.
9. J. Czyzowicz, L. Gasieniec, T. Gorry, E. Kranakis, R. Martin, and D. Pajak. Evacuating robots from an unknown exit located on the perimeter of a disc. In *DISC 2014*. Springer, Austin, Texas, 2014.
10. J. Dobbie. A survey of search theory. *Operations Research*, 16(3):525–537, 1968.
11. Y. Emek, T. Langner, J. Uitto, and R. Wattenhofer. Solving the ants problem with asynchronous finite state machines. In *Proceedings of ICALP, LNCS 8573*, pages 471–482, 2014.
12. P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous robots with limited visibility. *Theoretical Computer Science*, 337(1):147 – 168, 2005.
13. C. Lenzen, N. Lynch, C. Newport, and T. Radeva. Trade-offs between selection complexity and performance when searching the plane without communication. In *Proceedings of PODC*, pages 252–261, 2014.
14. A. López-Ortiz and G. Sweet. Parallel searching on a lattice. In *Proceedings of CCCG*, pages 125–128, 2001.
15. P. Nahin. *Chases and Escapes: The Mathematics of Pursuit and Evasion*. Princeton University Press, 2012.
16. L. Stone. *Theory of optimal search*. Academic Press New York, 1975.